PATENT ABSTRACTS OF JAPAN

(11)Publication number : 2002-189602
(43)Date of publication of application : 05.07.2002

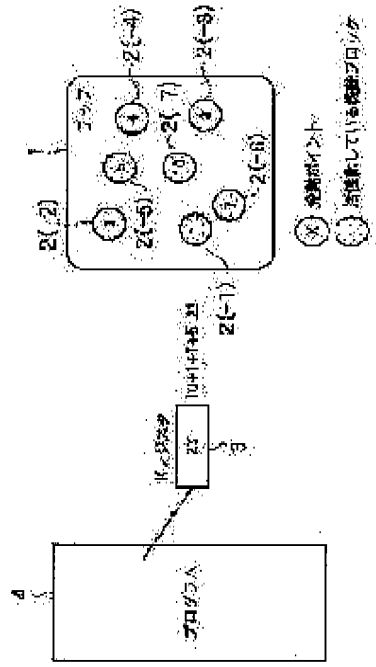
(51)Int.Cl. G06F 9/445
G06F 11/00

(21)Application number : 2000-390506
(22)Date of filing : 22.12.2000

(71)Applicant : TOSHIBA CORP
(72)Inventor : FUJIWARA TAKASHI
ASANO SHIGEHIRO
KUNIMATSU ATSUSHI

(54) PROCESSOR AND STORAGE MEDIUM FOR RECORDING HEAT GENERATION PREDICTOR CONTROL PROGRAM FOR PROCESSOR

(57)Abstract:
PROBLEM TO BE SOLVED: To provide a processor or controlling the amount of heat generated therein in harmony with its performance and a storage medium for recording a heat generation prediction control program for the processor.
SOLUTION: This processor has heating points correlated with the amount of heat generated therein provided one on each element or function block 2, and is structured so as to control the execution of the program by counting the heating points of activated elements or function blocks 2 to predict the amount of heat generated in a chip as a whole and the temperature of the chip.



(19)日本国特許庁（ＪＰ） (12) 公開特許公報（A） (11)特許出願公開番号
特開2002－189602
（P2002－189602A）
(43)公開日 平成14年７月５日(2002.7.5)

(51)Int.Cl.⁷	識別記号	ＦＩ	テーマコード（参考）
Ｇ０６Ｆ 9/445		Ｇ０６Ｆ 11/00	３５０Ｑ ５Ｂ０７６
11/00	３５０	9/06	６５０Ｂ

審査請求 未請求 請求項の数４ ＯＬ （全 ９ 頁）

(21)出願番号 特願2000－390506(P2000－390506)

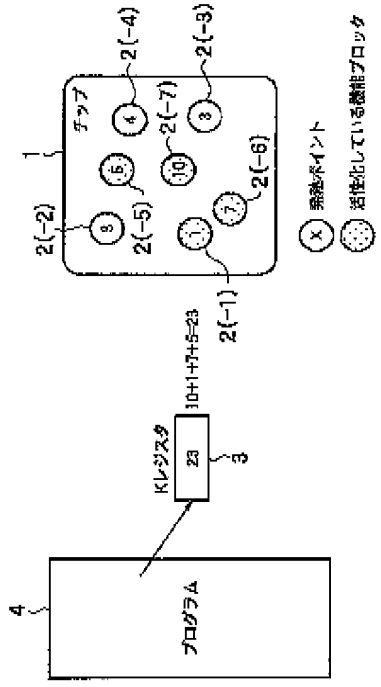
(22)出願日 平成12年12月22日(2000.12.22)

(71)出願人 000003078
株式会社東芝
東京都港区芝浦一丁目１番１号
(72)発明者 藤原 崇
神奈川県川崎市幸区小向東芝町１番地 株式会社東芝マイクロエレクトロニクスセンター内
(72)発明者 浅野 滋博
神奈川県川崎市幸区小向東芝町１番地 株式会社東芝研究開発センター内
(74)代理人 100083806
弁理士 三好 秀和 （外７名）

最終頁に続く

(54)【発明の名称】 プロセッサ及びプロセッサの発熱予測制御プログラムを記録した記録媒体

(57)【要約】
【課題】 この発明は、発熱量とパフォーマンスを調和させて制御するプロセッサ及びプロセッサの発熱予測制御プログラムを記録した記録媒体を提供することを課題とする。
【解決手段】 この発明は、素子または機能ブロック２毎に発熱量と相関関係のある発熱ポイントを設け、活性化している素子または機能ブロック２の発熱ポイントをカウントしてチップ全体の発熱量およびチップ温度を予測し、プログラムの実行を制御するように構成される。



1

【特許請求の範囲】

【請求項１】 機能ブロックが活性化されているか否かを検出する検出回路と、前記検出回路により前記機能ブロックの活性化が検出されて前記機能ブロックが発熱した時に、前記機能ブロックの発熱量に対応した発熱ポイントを出力する発熱ポイント出力レジスタとを備えた複数の機能ブロックと、

プロセッサ全体の発熱ポイントを保持する発熱ポイント保持レジスタと、

前記発熱ポイント保持レジスタに保持された値を、時間経過による放熱量に基づいて補正し、補正した値に活性化されている前記機能ブロックの発熱ポイント出力レジスタから出力された発熱ポイントを加え、前記発熱ポイント保持レジスタの値を更新する演算器とを具備し、

前記発熱ポイント保持レジスタに保持された発熱ポイントに基づいて現在のプロセッサの予測温度を求め、

以後実行可能な複数の特定の処理のいずれか１つの特定の処理を実行した場合のプロセッサの上昇温度を、前記特定の処理に対応した発熱ポイントを上昇温度に換算して求め、

前記予測温度に前記上昇温度を加えた値から、時間経過の放熱による下降温度を差し引いた値が、前記プロセッサの動作温度の許容制限値を越えない前記特定の処理が選択されて実行されることを特徴とするプロセッサ。

【請求項２】 プロセッサの単位時間あたりの放熱量に対応した枚数のチケットが単位時間毎に増加又は減少し、増加又は減少する前記チケットを保持するチケット保持レジスタと、

機能ブロックが活性化されているか否かを検出する検出回路と、前記検出回路により前記機能ブロックの活性化が検出されて前記機能ブロックが発熱した時に、前記機能ブロックの発熱量に対応した枚数のチケットを出力するチケット出力レジスタを備えた複数の機能ブロックと、

前記機能ブロックのチケット出力レジスタから出力されたチケットの枚数を加えて前記機能ブロックの総チケット数を求め、前記チケット保持レジスタの値から前記総チケット数を引き、引いた値に単位時間あたりのチケット増加分又は減少分を加えて前記チケット保持レジスタを更新する演算器とを具備し、

前記チケット保持レジスタに保持されたチケット枚数の範囲内で、処理に必要なチケット枚数がそれぞれ決められた複数の特定の処理の中から、前記プロセッサの動作温度の許容制限値を越えない１つの前記特定の処理が選択されて実行されることを特徴とするプロセッサ。

【請求項３】 機能ブロックが活性化されているか否かを検出する検出回路と、前記検出回路により前記機能ブロックの活性化が検出されて前記機能ブロックが発熱した時に、前記機能ブロックの発熱量に対応した発熱ポイントを出力する発熱ポイント出力レジスタとを備えた複

2

数の機能ブロックと、

プロセッサ全体の発熱ポイントを保持する発熱ポイント保持レジスタと、

前記発熱ポイント保持レジスタに保持された値を、時間経過による放熱量に基づいて補正し、補正した値に活性化されている前記機能ブロックの発熱ポイント出力レジスタから出力された発熱ポイントを加え、前記発熱ポイント保持レジスタの値を更新する演算器とを具備したプロセッサが実行する発熱予測制御プログラムを記録した記録媒体であって、

前記発熱ポイント保持レジスタに保持された発熱ポイントに基づいて現在のプロセッサの予測温度を求め、

以後実行可能な複数の特定の処理のいずれか１つの特定の処理を実行した場合のプロセッサの上昇温度を、前記特定の処理に対応した発熱ポイントを上昇温度に換算して求め、

前記予測温度に前記上昇温度を加えた値から、時間経過の放熱による下降温度を差し引いた値が、前記プロセッサの動作温度の許容制限値を越えない前記特定の処理を選択して実行させることを特徴とするプロセッサの発熱予測制御プログラムを記録した記録媒体。

【請求項４】 プロセッサの単位時間あたりの放熱量に対応した枚数のチケットが単位時間毎に増加又は減少し、増加又は減少する前記チケットを保持するチケット保持レジスタと、

機能ブロックが活性化されているか否かを検出する検出回路と、前記検出回路により前記機能ブロックの活性化が検出されて前記機能ブロックが発熱した時に、前記機能ブロックの発熱量に対応した枚数のチケットを出力するチケット出力レジスタを備えた複数の機能ブロックと、

前記機能ブロックのチケット出力レジスタから出力されたチケットの枚数を加えて前記機能ブロックの総チケット数を求め、前記チケット保持レジスタの値から前記総チケット数を引き、引いた値に単位時間あたりのチケット増加分又は減少分を加えて前記チケット保持レジスタを更新する演算器とを具備したプロセッサが実行する発熱予測制御プログラムを記録した記録媒体であって、

前記チケット保持レジスタに保持されたチケット枚数の範囲内で、処理に必要なチケット枚数がそれぞれ決められた複数の特定の処理の中から、前記プロセッサの動作温度の許容制限値を越えない１つの前記特定の処理を選択して実行させることを特徴とするプロセッサの発熱予測制御プログラムを記録した記録媒体。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】本発明は、発熱量及びチップ温度を予測し、予測をプログラムの実行に反映させたプロセッサ及びプロセッサの発熱予測制御プログラムを記録した記録媒体に関する。

【０００２】

【従来の技術】マイクロプロセッサに代表される半導体機器は近年の高周波数化に伴い、素子から発せられる発熱量が増大する傾向にある。素子の温度が上昇することで、一時的な動作不良あるいは半永久的な故障を引き起こすことがある。この問題を回避するために、従来では一般的に次のような対策がとられていた。

【０００３】（１）素子が最大の発熱をしたときにも温度が規定値以上に上がらないように機器に十分な放熱処理および放熱装置を付ける。

【０００４】（２）機器の放熱が一定という制約に基づいて、素子の動作周波数を下げる。

【０００５】（３）素子の温度変化を監視し、規定値以上に温度が上昇した場合にはダイナミックに動作周波数を下げる。

【０００６】一方、家庭用ゲーム機などで代表される一般消費者向けの半導体機器においては、高性能と同時に低価格であることが求められている。そこで、上記（１）の対策では十分な放熱を施すにはコストがかさみ、現実的ではない。また、上記（２）の対策では必要な性能が達成できない。

【０００７】さらに、動作周波数をダイナミックに変更する上記（３）の対策では、機器の放熱能力に応じた最大の周波数で動作せることができるという長所がある。しかしながら、この対策ではクリティカルな処理が必要なときに周波数が低下していたり、逆にクリティカルでないのに最大の周波数で動作していたりという、要求性能と動作周波数のミスマッチが起こる可能性がある。例えばゲーム機においては、動作周波数の低下が反応速度を悪化させたり、描画能力の低下により動画のこま落ちが発生する。また、素子の温度に基づいて動作周波数を変更した場合には、素子のばらつきにより素子毎に温度上昇が異なる可能性があり、製品毎の動作が異なることが考えられ、ゲーム機には向いていなかった。

【０００８】

【発明が解決しようとする課題】以上説明したように、低価格指向の半導体機器における従来の発熱対策にあっては、温度上昇に伴って動作周波数をダイナミックに変化させるといった対策が有効であった。しかし、このような対策にあっては、機器のパフォーマンスと動作周波数との間にミスマッチが発生するおそれがあり、十分なパフォーマンスを得ることができないといった不具合を招いていた。また、動作時の温度上昇の違いにより製品毎に動作が異なってしまうおそれもあった。

【０００９】そこで、この発明は、上記に鑑みてなされたものであり、その目的とするところは、発熱量とパフォーマンスを調和させて制御するプロセッサ及びプロセッサの発熱予測制御プログラムを記録した記録媒体を提供することにある。

【００１０】

【課題を解決するための手段】上記目的を達成するために、課題を解決するための第１の手段は、機能ブロックが活性化されているか否かを検出する検出回路と、前記検出回路により前記機能ブロックの活性化が検出されて前記機能ブロックが発熱した時に、前記機能ブロックの発熱量に対応した発熱ポイントを出力する発熱ポイント出力レジスタとを備えた複数の機能ブロックと、プロセッサ全体の発熱ポイントを保持する発熱ポイント保持レジスタと、前記発熱ポイント保持レジスタに保持された値を、時間経過による放熱量に基づいて補正し、補正した値に活性化されている前記機能ブロックの発熱ポイント出力レジスタから出力された発熱ポイントを加え、前記発熱ポイント保持レジスタの値を更新する演算器とを具備し、前記発熱ポイント保持レジスタに保持された発熱ポイントに基づいて現在のプロセッサの予測温度を求め、以後実行可能な複数の特定の処理のいずれか１つの特定の処理を実行した場合のプロセッサの上昇温度を、前記特定の処理に対応した発熱ポイントを上昇温度に換算して求め、前記予測温度に前記上昇温度を加えた値から、時間経過の放熱による下降温度を差し引いた値が、前記プロセッサの動作温度の許容制限値を越えない前記特定の処理が選択されて実行されることを特徴とする。

【００１１】第２の手段は、プロセッサの単位時間あたりの放熱量に対応した枚数のチケットが単位時間毎に増加又は減少し、増加又は減少する前記チケットを保持するチケット保持レジスタと、機能ブロックが活性化されているか否かを検出する検出回路と、前記検出回路により前記機能ブロックの活性化が検出されて前記機能ブロックが発熱した時に、前記機能ブロックの発熱量に対応した枚数のチケットを出力するチケット出力レジスタを備えた複数の機能ブロックと、前記機能ブロックのチケット出力レジスタから出力されたチケットの枚数を加えて前記機能ブロックの総チケット数を求め、前記チケット保持レジスタの値から前記総チケット数を引き、引いた値に単位時間あたりのチケット増加分又は減少分を加えて前記チケット保持レジスタを更新する演算器とを具備し、前記チケット保持レジスタに保持されたチケット枚数の範囲内で、処理に必要なチケット枚数がそれぞれ決められた複数の特定の処理の中から、前記プロセッサの動作温度の許容制限値を越えない１つの前記特定の処理が選択されて実行されることを特徴とする。

【００１２】第３の手段は、機能ブロックが活性化されているか否かを検出する検出回路と、前記検出回路により前記機能ブロックの活性化が検出されて前記機能ブロックが発熱した時に、前記機能ブロックの発熱量に対応した発熱ポイントを出力する発熱ポイント出力レジスタとを備えた複数の機能ブロックと、プロセッサ全体の発熱ポイントを保持する発熱ポイント保持レジスタと、前記発熱ポイント保持レジスタに保持された値を、時間経過による放熱量に基づいて補正し、補正した値に活性化

されている前記機能ブロックの発熱ポイント出力レジスタから出力された発熱ポイントを加え、前記発熱ポイント保持レジスタの値を更新する演算器とを具備したプロセッサが実行する発熱予測制御プログラムを記録した記録媒体であって、前記発熱ポイント保持レジスタに保持された発熱ポイントに基づいて現在のプロセッサの予測温度を求め、以後実行可能な複数の特定の処理のいずれか１つの特定の処理を実行した場合のプロセッサの上昇温度を、前記特定の処理に対応した発熱ポイントを上昇温度に換算して求め、前記予測温度に前記上昇温度を加えた値から、時間経過の放熱による下降温度を差し引いた値が、前記プロセッサの動作温度の許容制限値を越えない前記特定の処理を選択して実行させることを特徴とする。

【００１３】第４の手段は、プロセッサの単位時間あたりの放熱量に対応した枚数のチケットが単位時間毎に増加又は減少し、増加又は減少する前記チケットを保持するチケット保持レジスタと、機能ブロックが活性化されているか否かを検出する検出回路と、前記検出回路により前記機能ブロックの活性化が検出されて前記機能ブロックが発熱した時に、前記機能ブロックの発熱量に対応した枚数のチケットを出力するチケット出力レジスタを備えた複数の機能ブロックと、前記機能ブロックのチケット出力レジスタから出力されたチケットの枚数を加えて前記機能ブロックの総チケット数を求め、前記チケット保持レジスタの値から前記総チケット数を引き、引いた値に単位時間あたりのチケット増加分又は減少分を加えて前記チケット保持レジスタを更新する演算器とを具備したプロセッサが実行する発熱予測制御プログラムを記録した記録媒体であって、前記チケット保持レジスタに保持されたチケット枚数の範囲内で、処理に必要なチケット枚数がそれぞれ決められた複数の特定の処理の中から、前記プロセッサの動作温度の許容制限値を越えない１つの前記特定の処理を選択して実行させることを特徴とする。

【００１４】

【発明の実施の形態】以下、図面を用いてこの発明の一実施形態を説明する。

【００１５】本発明のプロセッサは、チップが大規模、高周波数化するにつれて問題となる発熱を考慮し、プログラム的に並列度が余りなくクリティカルな部分でない時は発熱を抑え、並列度が大きくプログラム的にクリティカルとなる部分では最高のパフォーマンスを出せるように、発熱をソフトウェアから制御可能としたものである。

【００１６】本発明では、チップ内部の活性化している素子または機能ブロック毎に発熱量に応じた発熱ポイントを設定する。発熱量そのものを発熱ポイントとしてもよく、また全ての素子の発熱量が等しければ、チップ活性化率で代用することもできる。命令実行中に活性化している部分の発熱ポイントをカウントすることにより、ある時点でのチップ全体の発熱ポイントを算出する。素子や機能ブロック等の発熱ポイントは設計時にＣＡＤツールなどで見積もった発熱量を基に決定する。このカウントしたチップの発熱ポイントをプログラムの動作にフィードバックすることで不要な発熱を抑え、クリティカルな処理が必要なときに周波数が低下することがないようにする。

【００１７】ここで、素子の直接的な温度ではなく、素子または機能ブロックの活性化状態に応じた発熱ポイントを用いることにより、異なるチップであっても同じプログラムであれば同じ動作をすることが保証され、実際の温度を測定しそれをプログラムにフィードバックした場合のようにチップ毎のばらつきによる動作の違いは発生しなくなる。

【００１８】プログラムは、チップの発熱ポイントまたはこれを基に予測されたチップ温度を読み出すことにより、機器の性能と発熱量のトレードオフを選択できるようにする。このように、チップの発熱量をプログラムから知ることができる手段を設けることにより、プログラムのクリティカルでない部分では発熱を十分抑えたプログラムを実行し、クリティカルな部分では発熱と性能のトレードオフを考慮したプログラムを実行するようにする。

【００１９】図１はこの発明の一実施形態に係るプロセッサの発熱ポイントをプログラムが認識する様子を示す図である。

【００２０】図１において、上述したように、チップ１の機能ブロック２毎に発熱ポイントを設ける。この発熱ポイントは、上述したようにプロセッサの設計時にＣＡＤやシミュレーション等により見積もり評価して予め用意する。例えば図１において機能ブロック２（-1）の発熱ポイントは「１」、機能ブロック２（-2）及び機能ブロック２（-3）の発熱ポイントは「３」、機能ブロック２（-4）の発熱ポイントは「４」、機能ブロック２（-5）の発熱ポイントは「５」、機能ブロック２（-6）の発熱ポイントは「７」、機能ブロック２（-7）の発熱ポイントは「１０」とする。このように設定された発熱ポイントを例えばメモリマップしたレジスタ３（以下、Ｋレジスタと呼ぶ）に割り付けておき、プログラム４から読めるようにする。図１では、例えば機能ブロック２（-1）、（-5）、（-6）、（-7）が活性化されていると、それぞれの機能ブロック２の発熱ポイントが加算されて、加算値２３（＝１＋５＋７＋１０）がＫレジスタ３に保持される。

【００２１】このＫレジスタ３の値はチップ１の放熱に相当し、規定した周囲の温度（常温）とＫレジスタ３の値の差に比例した割合で値（温度）を減少させる。素子の放熱では温度が下がっても周囲温度まで下がるのが限界なので、周囲温度に相当するＫレジスタ３の下限値を

設け、この下限値に達した場合はＫレジスタ３の値の減少を中止する。また、予測温度がある一定以上に達した場合には動作周波数を低下させる機構を設けておく。

【００２２】図２は図１に示すＫレジスタ３の内容を更新する構成を示す図である。図２において、更新に必要な構成として、減算器２１、加算器２２を有し、それぞれの機能ブロック２は、活性化検出回路２３ならびに発熱ポイントレジスタ２４を備えている。減算器２１は、現在のＫレジスタ３の値からチップ全体の単位時間あたりに減少する発熱ポイントを減じる回路である。単位時間あたりに減少する発熱ポイントは、現在のＫレジスタ３の値と外界の温度により決定される値、もしくはある決められた固定値とする。

【００２３】各機能ブロック２に備えられた活性化検出回路２３は、機能ブロック２が活性化しているか否かを判定する回路である。また、活性化検出回路２３は機能ブロック２が活性化していると判定した場合は、発熱ポイントレジスタ２４に格納されている対応する機能ブロック２の発熱ポイントを加算器２２に出力する。加算器２２は、活性化している機能ブロック２の発熱ポイントレジスタ２４から出力された発熱ポイントと、現在のＫレジスタ３の値を補正した値、すなわち減算器２１の出力を加算し、加算値をＫレジスタ３の更新値とする。

【００２４】図３は活性化している機能ブロックを検出する検出例を示す図である。この例では、機能ブロックとなる例えば各コプロセッサ３１は、コアモジュール３２と結合しており、コアモジュール３２が各コプロセッサ３１を必要に応じて起動する。起動されたコプロセッサ３１はコアモジュール３２から与えられたデータを使い計算し、結果をコアモジュール３２に与える。この場合に、コアモジュール３２はコプロセッサ３１を起動した時に、各コプロセッサ３１に備えられたビジーレジスタ３３のビジーフラグを立てる。ビジーフラグは、コアモジュール３２がコプロセッサ３１から処理結果を与えられた時、もしくはコプロセッサ３１がコアモジュール３２に処理結果を与えた時に落とされる。図２に示す活性化検出回路２３は、このビジーフラグを検出することにより各機能ブロックのコプロセッサ３１の活性化状況を判定する。

【００２５】このようにして得られるチップ１の発熱ポイントに基づいて、チップ１の温度上昇を予測する。プログラム４がこのチップ温度を知ることでプログラムの動作を変更し、処理量を可変にすることで発熱を制御する。例えば、プログラム４のクリティカルな部分の直前で温度が規定した値に近づいていることがわかれば、画像の質を落とすなどして処理量を減らし、温度上昇を抑えることでクリティカルな部分でも素子の温度が規定値より上がらないようにする。

【００２６】次に、プログラムが素子又は機能ブロック２の温度を知ることによってどのように制御を行うのか、具体例を挙げて説明する。

【００２７】一般に発熱は、プロセッサで行われる様々なアクティビティによって変動する。理想的には、プロセッサで行われる様々な命令毎に機能ブロック２や素子の発熱ポイントを求め、この発熱ポイントに基づいて各命令の発熱量を予測して温度上昇が見積もれるようにすることである。しかし、この方法では発熱量を見積もるために命令毎に発熱量の加算を行わなければならず、効率がよくない。そこで、発熱量を見積もる単位を命令毎ではなく、プログラムの中の関数コールや、サブルーチンレベルで行うことで発熱量の加算の手間を減らすことができる。この方法では、予めシミュレーションなどにおいてプログラム中の関数やサブルーチンが実行された時に活性化される機能ブロック２や素子の発熱ポイントを算出し、プログラムの実行時にこの見積もりに見合った処理を行うことで温度上昇を一定に抑える。

【００２８】例えばプロセッサが以下に示すような状態にあるとする。

【００２９】（１）現在のチップ１の温度が１１０℃で、１１５℃までが許容される動作範囲である。

【００３０】（２）単位時間当たりのチップ１の発熱ポイントが４０の場合にはチップ１の温度は３℃上昇し、発熱ポイントが６０の場合にはチップ１の温度は５℃上昇し、発熱ポイントが８０の場合にはチップ１の温度は８℃上昇する。

【００３１】（３）周囲温度を２５℃と考え、これから実行しようとするサブルーチンＡ，Ｂ，Ｃの違いによる温度低下の割合は誤差範囲と仮定し、単位時間当たり１℃チップ温度が下がる。

【００３２】このような状態において、これから呼び出そうとするサブルーチンにサブルーチンＡ，Ｂ，Ｃの３つの選択肢があったとする。サブルーチンＡの単位時間当たりのチップ発熱ポイントは４０、サブルーチンＢのそれは６０、サブルーチンＣのそれは８０とすると、それぞれのサブルーチンＡ，Ｂ，Ｃを実行した場合のチップ１の予測温度は、以下に示すようになる。なお、各サブルーチンの発熱ポイントは予めシミュレーションなどで求めておく。

【００３３】サブルーチンＡ＝１１０（現在のチップの温度）＋３（発熱ポイントに基づいて換算した上昇温度）－１（放熱による下降温度）＝１１２＜１１５（許容動作温度）

サブルーチンＢ＝１１０（同上）＋５（発熱ポイントに基づいて換算した上昇温度）－１（同上）＝１１４＜１１５（許容動作温度）

サブルーチンＣ＝１１０（同上）＋８（発熱ポイントに基づいて換算した上昇温度）－１（同上）＝１１７＞１１５（許容動作温度）

このような処理のプログラム例を図４に示す。図４において、

LIMITTEMP ：許容動作温度
RISETEMPA ：サブルーチンＡの発熱ポイント
RISETEMPB ：サブルーチンＢの発熱ポイント
RISETEMPC ：サブルーチンＣの発熱ポイント
Point2Temp() ：発熱ポイントを温度に変換する関数
GetKReg() ：Ｋレジスタ３の値を得る関数
EstimateP2T() ：各機能ブロックの発熱ポイントを温度に変換し、かつ現在のチップの予測温度から放熱温度を見積もる関数

である。上記Point2Tempの関数により現在のチップの予測温度が認識でき、EstimateP2T の関数により各機能ブロック２の発熱ポイントが温度に換算され、さらに現在のチップ温度から放熱温度が見積もられる。

【００３４】このようなプログラムを実行して得られる予測結果から、実行できるサブルーチンは、サブルーチンＡまたはＢであることがわかる。これにより、チップの温度が動作許容範囲内に収まる範囲でプロセッサが最大の性能を達成するように次に実行する処理を選択するようにすればよい。

【００３５】従って、このような実施形態においては、直接測定したチップの温度を用いるのではなく、発熱ポイントを用いることにより、チップのばらつきによるプログラム動作の違いを無くすことができる。また、チップの素子や機能ブロックの活性化の度合をプログラムにフィードバックすることにより不要な発熱を抑制することが可能となり、プログラムでクリティカルな処理が必要な場合には動作周波数が低下することがないように制御することが可能となる。

【００３６】次に、この発明の他の実施形態を説明する。

【００３７】この実施形態の特徴とするところは、温度を直接プログラムに見せるのではなく、温度と相関をもったチケットという概念を導入したことにある。チケットは、例えばレジスタ（以下、Ｔレジスタと呼ぶ）に保存され、このＴレジスタはプログラムから見えるようにしておく。ある目標とする定常状態の温度（Ｔｓ）での単位時間当たりの放熱量ＱＴｓとバランスした発熱ポイントに基づいて発行するチケット数ｔを求める。チケット数ｔを全て消費した場合には、放熱量ＱＴｓの熱を発生するが、放熱とのバランスが取れてるので温度Ｔｓ以上にはならないことが保証される。チケットと発熱ポイントの間は比例関係にあればよいので、ここでは、チケット数＝発熱ポイントと考える。

【００３８】チケットは、単位時間毎に決められた枚数（ここではｔ枚）増加し、命令列を実行する場合にそれに対応したチケットを所得して実行を行い、チケット数はその分だけ自動的に減少する。ただし、チケットを節約して実行しているとチケット枚数がどんどん増えてしまい、大量に一度にチケットを使用して命令列を実行してしまうと、制限している温度を一時的に越えてしまうことが考えられる。そこで、チケットの保持できる枚数は一度にチケットを大量に使用しても、制限温度を越えないようなチケット枚数を上限値とする。

【００３９】また、発熱ポイントとチップ温度との相関関係が正しくなく予想温度と実際の温度がずれている場合に備え、実際のチップ温度も測定しておき、チップの温度が上限値を越えてしまった場合には、動作周波数を落としてチップ温度を下げる機能、もしくはチップの動作そのものを停止する機能を備えている。リセットがかかった場合には、Ｔレジスタの値は０になる。Ｔレジスタの値は一般のプログラムからは設定できないようにして、プログラマの不注意もしくは故意に変更できない。例えば、Ｔレジスタはスーパーバイザモードでしか変更できず、一般のアプリケーションプログラムで設定できないようにすることは容易に実現できる。

【００４０】このようにチケットにより発熱量を制限することで温度をある一定以上にならないようにし、プログラムの並列度があまりなくクリティカルでない部分ではチケットを温存し、並列度が多くあるプログラムの部分やクリティカルな部分では温存していたチケットをフルに使い多くの機能ブロックを使って処理を行うことが可能になる。なお、Ｔレジスタの値は、単位時間毎に減少するようにし、各機能ブロックや素子が動作するとその発熱量に相当したチケット枚数が自動的に増加し、減少の割合を周囲温度を一定として目標とした温度での放熱量と等しい熱を発生する分のチケット枚数にしてチケットの増減を反転し、指定した温度以上にならないよう制御するようにしてもよい。この場合に、Ｔレジスタの下限値は０とし、リセット時には一度にチケットを発行しても規定温度を超えない枚数とする。

【００４１】図５は上記Ｔレジスタの値を更新する構成を示す図である。図５において、各機能ブロック２の検出回路５１は、図２に示す検出回路２３と同様に機能ブロック２の活性化を検出し、ＦＴレジスタ５２に保存されている発熱ポイントと相関を持った実行に必要なチケット数を出力する回路である。各機能ブロック２の消費チケット数を加算器５３で加算して総チケット数を求める。この加算値が、実行にかかった総チケット数である。現在のＴレジスタ５４の値からこの総チケット数を減算器５５を用いて引き、加算器５６により単位時間あたりのチケット増加分を加えたものが、次のＴレジスタ５４の値となる。

【００４２】次に、上記チケットを採用した場合のプログラムの制御例を図６を参照して説明する。なお、図６において、CurTicket() は、Ｔレジスタの値を参照する関数であり、GetTicket() は、最初の項目が関数を実行するために必要なチケットの枚数であり、次の項目がしきい値となる。

【００４３】図６において、現在のチケット数は２００とし、単位時間あたりのチケットの増加を６０とし、関

数FuncA及び関数FuncBの実行には２単位時間かかるものとする。従って、FuncA及びFuncBの実行後にはチケットは１２０増加することになる。FuncA及びFuncBは同様な処理を行う関数とし、例えばFuncBはFuncAよりも処理が軽いが画質が多少落ちる関数である。FuncA、FuncB、CriticalFuncはそれぞれチケットを１４０、６０、３００消費するものとする。図６に示すプログラム例では、Point Aでの選択例を示し、いずれの選択例においても同様な処理を行っている。Point AでのCurTicket() の戻り値は、２００＋６０＝２６０である。CriticalFuncを実行するにはチケットを３００必要とするので、Point AではFuncBを実行することになる。

【００４４】このように、上記実施形態においても、先の実施形態で得られる効果と同様の効果を得ることができる。

【００４５】

【発明の効果】以上説明したように、この発明によれば、素子または機能ブロック毎に発熱量と相関関係のある発熱ポイント又はチケットを設け、活性化している素子または機能ブロックの発熱ポイント又はチケットに基づいてチップ全体の発熱量およびチップ温度を予測し、プログラムの実行を制御するようにしたので、チップのばらつきによるプログラム動作の違いをなくし、かつ不要な発熱を抑制し、クリティカルな処理が必要なときに動作周波数が低下することを防止することができる。

【図面の簡単な説明】

＊【図１】この発明の一実施形態に係るプロセッサの発熱ポイントをプログラムが認識する様子を示す図である。

【図２】Ｋレジスタを更新する構成を示す図である。

【図３】活性化している機能ブロックを検出する検出例を示す図である。

【図４】この発明の一実施形態に係るプログラム例を示す図である。

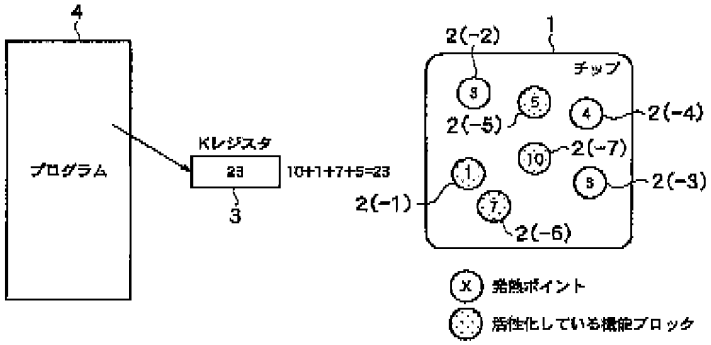
【図５】この発明の他の実施形態に係るプロセッサのＴレジスタを更新する構成を示す図である。

【図６】この発明の他の実施形態に係るプログラム例を示す図である。

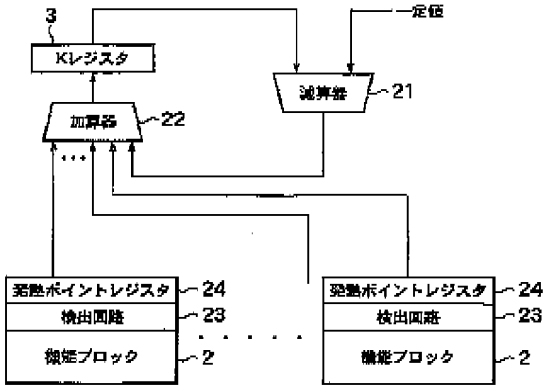
【符号の説明】

１　チップ
２　機能ブロック
３　Ｋレジスタ
４　プログラム
２１，５５　減算器
２２，５３，５６　加算器
２３，５１　検出回路
２４　発熱ポイントレジスタ
３１　コプロセッサ
３２　コアモジュール
３３　ビジーレジスタ
５２　ＦＴレジスタ
５４　Ｔレジスタ

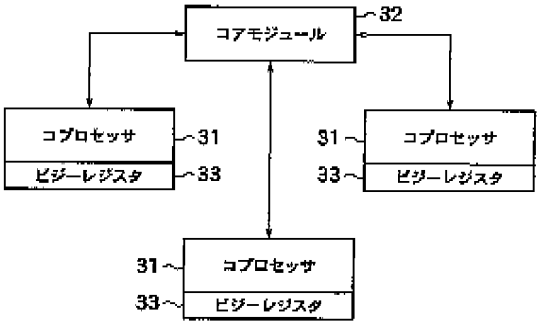
【図１】



【図２】



【図３】

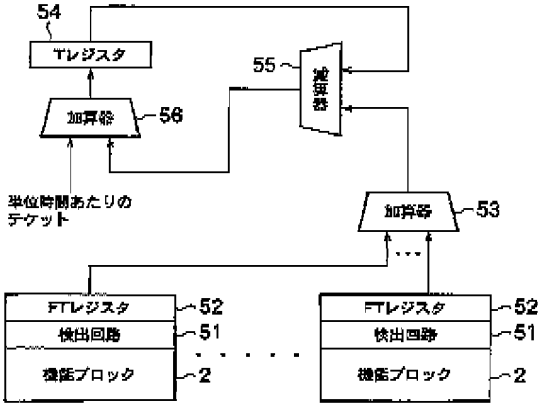


【図４】

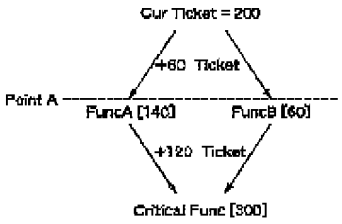
```
#define LIMITTEMP 115
#define RISETEMPA 40
#define RISETEMPB 60
#define RISETEMPC 80


int Cur_Temp = Point2Temp(GetKReg ());
if (LIMITTEMP > Cur_Temp + EstimateP2T (Cur_Temp, RISETEMPC)
   SubroutineC ();
else if (LIMITTEMP > Cur_Temp + EstimateP2T(Cur_Temp, RISETEMPB)
   SubroutineB ();
else if (LIMITTEMP > Cur_Temp + EstimateP2T(Cur_Temp, RISETEMPA)
   SubroutineA ();
```

【図５】



【図６】



Point A でのプログラム例

```
if (Cur Ticket () + 120 > 300 + 140 ) {
   Get Ticket (140) ;
   Func A () ;
} else if (Cur Ticket () + 120 > 300 + 60 ) {
   Get Ticket (60) ;
   Func B () ;
}
```

OR

```
if (Get Ticket (140, 300-120))
   Func A () ;
else if (Get Ticket (60, 300-120))
   Func B () ;
```

フロントページの続き

(72)発明者 国松 敦
神奈川県川崎市幸区小向東芝町１番地 株式会社東芝マイクロエレクトロニクスセンター内

Ｆターム(参考) 5B076 AB17